

浙江大学

本科生毕业论文(设计)



题目 基于深度学习的密码猜测算法研究

姓名与学号 刘一璟 3150105531

指导教师 冯涛

年级与专业 15级数学与应用数学

所在学院 数学科学学院

提交日期 2019.05.26

目 录

第一部分 毕业论文（设计）

1 引言	3
1.1 背景与相关工作	4
2 正文	7
2.1 方法	7
2.1.1 对抗生成网络	7
2.1.2 TPGGAN	8
2.2 实验	10
2.2.1 数据集	10
2.2.2 实验组成	10
2.2.3 非定向攻击	11
2.2.4 定向攻击	12
2.3 评论	14
3 结论	16
4 参考文献	17

第一部分

毕业论文（设计）

摘 要

由于硬件与算法的发展，以及严重的信息泄漏问题，认证系统的安全性现今成为一个值得关注的问题。现有的密码猜测算法，能够在离线攻击中以极高的效率对密码哈希进行修复，它们通常能够在短时间内生成大量的高质量候选密码，并快速地与密码哈希群体进行匹配。这些方法通常基于一个富有代表性的字典，即一个较长的字符列表，并结合一些高效的生成规则来进行字典扩展，在实践中拥有出色的猜测性能，然而其缺陷也是明显的，这些方法来源于人工分析与长时间的实践测试，因此如果想要对其性能进一步改进，一般无法在短期内完成。

在这篇文章中，我们以一种完全不同的思路构建了新的密码猜测算法，称为 TPGGAN。这项工作基于深度学习框架，无须任何形式有关密码的先验知识，从密码数据集本身学习其内在特征，并自动地完成算法构建过程。TPGGAN 属于对抗生成模型，它对高斯分布到真实密码分布的变换进行建模，通过对变换后的分布进行采样的方式，生成大量的高质量候选密码。同时，我们的方法考虑了对特定用户群体进行攻击的场景，能够充分利用获取的额外用户信息，大幅度地提高密码猜测效率，较好地完成定向攻击任务。我们的实验结果表明 TPGGAN 是一个相当有竞争力的算法，通过在两个大型密码数据集上验证我们的方法，TPGGAN 能够达到现有最好方法的匹配率，并且与其它方法结合后，可以进一步提升其它方法的表现。在定向攻击场景中，实验结果也表明了当提供一部分密码信息时，TPGGAN 相较非定向攻击高于 35% 的效率进行密码猜测。

关键词：深度学习，密码猜测，定向密码猜测

Abstract

The safety of authorization system becomes a dominant issue due to the development of hardware, algorithm and acute data breaches. State-of-the-art password cracking methods can discover password hashes with extreme efficiency, they are commonly able to generate lots of high-quality candidate passwords and compare theirs hashes with the leaks. These approaches are generally based on a representative dictionary, i.e., a long list composed of character strings, they extend this dictionary using some effective generation rules. Although performing well on guessing task, they are obviously limited, both dictionary and generation rules are updated by expertise and practice, which leads to a remarkable obstacle if one requires further powerful tools, it's time-consuming to improve them.

In this paper, we construct a novel cracking algorithm from different points of view, which is called TPGGAN. This work is based on the framework of deep learning, which requires zero prior knowledge of password, it models the transformation from gaussian distribution to the true password distribution and samples from the transformed distribution to generate high-quality candidate passwords. Moreover, we take the attack to specific population of users into consideration, our method gets more efficient if extra information of users is provided, which shows strongly competitive in the task of targeted password guessing. our experimental results prove that TPGGAN is a highly competitive algorithm, we evaluate it on two large-scale password datasets, our approach achieves the match rate of state-of-the-art methods and improves other methods when it is combined with these methods. In the circumstance of targeted password cracking, the experiments show that if additional information of password is available, TPGGAN can crack with a higher efficiency which is bigger than 35%.

Keywords: Deep Learning, Password Guessing, Targeted Password Guessing

1 引言

文本形式的密码现今仍然是主流的认证方式，这主要来源于它的简易性、优秀的可实现性等特点，在未来的较长一段时间中，这种方式将会被持续使用。但在用户设计密码时，通常为了便于记忆，会倾向于使用易于破解的密码。这就导致了允许用户自定义密码的认证系统在面对空间缩减攻击时表现得更为脆弱，可以在远低于穷举攻击的难度下被攻破 [Yampolskiy, 2006](#)。

密码猜测工具提供了一个常用的用于评估用户所使用密码强度的方式 [Dell'Amico, Michiardi, and Roudier, 2010; Kelley et al., 2012](#)，这对提升认证系统的安全性有着指导意义。通常密码猜测算法不会使用穷举搜索的方式进行候选密码的生成，而是使用从公开或者泄露的密码样本与字典中选取单词。在这个过程中，诸如 HashCat [HashCat 2017](#) 以及 John the Ripper [John the Ripper 2017](#) 等密码猜测工具会使用一些启发式的密码变换，这些变换模仿了用户设计密码的习惯，结合马尔科夫模型，可以生成大量相似的密码猜测样本。

在这项工作中，我们采用了完全不同的方式。尽管上述的密码猜测工具通常在实际应用中拥有不错的表现，但其对密码样本的利用仅限于进行一些启发式的变换，并没有从密码样本群体抽取其本身的一些特质，即并非通过近似密码样本的分布来进行生成。基于深度学习的方法，我们提出了一个新的密码猜测算法，从公开的密码样本点中还原真实的密码分布，并通过对学习到的真实分布进行采样进行候选密码的生成，避免了对密码数据集的分析以及启发式变换的设计，完全自治地从数据集中确定算法的运作方式。

我们的方法称为 TPGGAN (Targeted Password Guessing GAN)，它使用条件对抗生成网络 (conditional generative adversarial nets, conditional GAN) 从密码数据集的一部分建模密码的真实分布，并在剩余的一部分上进行测试。在两个大型密码数据集上的结果表明，所提出的方法能够超过现有基于规则以及机器学习的密码猜测工具，尽管我们使用的方法没有使用任何关于密码的先验知识。如果对 TPGGAN 的输出使用 HashCAT，那么相比单独使用 HashCAT 产生的结果将可以匹配 54%-72% 更多的测试集中的密码，这表明了 TPGGAN 确实能够抽取到以往工具无法捕捉的密码性质。同时，TPGGAN 对以往方法无法进行定向攻击的问题进行了改进，在拥有定向攻击目标一部分信息的情况下，TPGGAN 的破解效率将提升 35% - 61%。

1.1 背景与相关工作

密码猜测攻击有多种形式，在离线密码修复中，攻击者通常只拥有原密码的哈希形式。在一次攻击尝试中，攻击者生成一个候选密码，然后使用合适的密码哈希算法得到候选密码的哈希值，如果这个哈希值与原密码的哈希匹配，那么即完成了一次成功的修复。上述是密码猜测攻击最为简单的形式，在这项工作中，我们主要讨论这种情况。

在候选密码的生成方式上，有两种最为常见的方式。暴力破解是最为直接的一种，在这种方式下，攻击者将会遍历每一种可能的组合，作为一次候选密码生成。暴力破解将会遍历整个字符空间，因此密码修复必然成功，但由于过于低效，该方法在实际使用中并不可行。如果没有使用混淆方法，那么通过时间与存储的交换以及预计算的技术，暴力破解将可以被很大幅度地改进。

另一种攻击方式被称为字典攻击，字典本身构成了一个词汇列表，通常被认为是用户选择便于记忆的密码的主要来源。然而，用户并不会直接使用上面的词汇，而是将其进行一定程度的调整，使得其仍然容易记忆。在字典攻击中，攻击者通常会尝试复原用户使用字典词汇设计密码的过程，从给定字典中挑选词汇并使用预先选定的变化规则来自动地生成变体。对于一次成功的字典攻击，需要字典中包含有用户使用的词汇，并且攻击者选定了正确的变换规则。通常来说，字典攻击要比暴力破解快很多，然而攻击者仍然受限于时间限制下可以使用的变换规则数量，当给定字典的大小增大时这种限制将会变得相当严重，所以需要更好地筛选使用的变换规则。

John the Ripper 和 HashCat John the Ripper(JtR) 与 HashCat 是最为著名密码破解工具，它们拥有很多种生成密码的方式。如穷举攻击；字典攻击；规则攻击，规则定义了如何将字典中的词进行变换产生新的候选密码。JtR 与 HashCat 也支持马尔可夫模型的候选密码生成，与规则攻击结合可以相当有效地进行离线密码修复。

基于 Markov 模型的密码猜测算法 马尔可夫模型在计算机安全尤其在密码安全中是十分有效的。在密码破解中它拥有优秀的表现，同样也可以用于评估用户所选用密码的安全强度。最近的工作比较了许多形式的概率密码模型，结果上表明相较概率上下文无关语法(Probabilistic Context Free Grammar) 来说马尔可夫模型能更好地模型化密码分布。

Claude 等 [Castelluccia, Dürmuth, and Perito, 2012](#) 使用了全字符串的马尔可夫模型以进行密码安全强度评估，并提出了适应于不同数据集的适应性密码强度度量函数，通过增加一些预先计算得到的噪声，可以保证其构造的马尔可夫模型对于 n-gram 数据库泄漏而言是安全的。

Arvind 等 [Narayanan and Shmatikov, 2005](#) 利用基于马尔可夫模型的马尔可夫滤波器来

捕捉语音上的相似性，指出人类使用非字母符号的模式可以被有限自动机模型化，作者还提出两个算法用于高效统计符合马尔可夫滤波器的定长字符串以及高效统计一个确定有限自动机接受的定长字符串，通过进行时间空间上的交换来进行快速的密码破解。

基于概率上下文无关语法的密码猜测算法 概率上下文无关语法 (probabilistic context-free grammars) 简称 PCFG，它被应用于密码猜测算法中是由于典型密码通常拥有确切的结构，不同结构的分布可以通过样本来抽取，在之后这些结构将被用于产生候选密码猜测。

Matt 等 [Weir et al., 2009](#) 使用了 PCFG 从密码训练集中建模不同密码模式的差异性，首次通过利用大规模真实密码样本作为训练数据来自动地得到用户设计密码的模式。从一个密码样本构成的训练集中自动地产生一个 PCFG，并以此得到一些密码变换规则用于密码猜测的候选生成过程中。

Jerry 等 [Ma et al., 2014](#) 在 Matt 等的工作上进行了扩展，Matt 等的算法首先需要从训练集学习不同模版的分布，然后通过一个选定的字典来实例化候选密码猜测，Jerry 等尝试了三种不同的字典，最终的结果表明通过训练集生成的字典拥有最好的表现。

基于深度学习的密码猜测算法 深度学习提供了拥有出色性能的函数近似工具，被广泛地使用于多种任务中，在不同的应用场景中都表现出了领先的性能，近年来有一些基于它的密码猜测算法被提出。

最早的工作可以追溯至 2006 年，Ciaramella 等 [Ciaramella et al., 2006](#) 等讨论分析了基于多层感知机的主动密码检查器，并验证了不同拓扑结构神经网络的表现，实验结果表明 Ciaramella 等的方法在当前可选方法中是一个合适的替代方法，并且在资源受限的设备上表现出了相对最好的性能。

Melicher 等 [Melicher et al., 2016](#) 提出了 FLA，基于循环神经网络 (recurrent neural networks) 的密码猜测算法，用于解决密码强度评估问题。他们的工作表明了神经网络通常能够比现有最好方法更有效率地进行密码猜测，如概率上下文无关语法和马尔可夫模型等，神经网络也可以被高度压缩并且不会明显地降低猜测性能。

Briland 等 [Hitaj et al., 2017](#) 提出了 PassGAN，通过对抗神经网络 (Generative Adversarial Network) 从真实的密码数据集中自治地学习对应分布，并以此生成高质量的密码猜测。PassGAN 避免了密码的先验知识与常见样式，能够达到与现有最好方法相近的结果，在一些情况下拥有更好的表现。

总结 基于启发式规则的字典攻击，其规则变换通常是点对点的，尝试复原用户设计密码的过程，而不是源于对大规模密码数据集的分析得到的。启发式的规则局限于获得密码空

间的特定子集，这些子集的好坏取决于规则是否有效，并且以人工形式构建和测试启发式规则是一个低效的过程，这限制了算法的更新迭代过程。

基于 PCFG 与马尔可夫模型的算法都可以利用密码数据集来抽取密码的结构，这些方法通常更加自动化，减少了对先验知识的依赖，但对密码分布本身有着较强的假设，如基于马尔可夫模型的算法隐式地假设了密码特征可以由 n-grams 来进行定义，PCFG 方法对单个密码不同部分有着较强的概率无关性假设，这些因素使得算法拥有较大的局限性。

基于深度学习的算法对密码分布本身没有任何形式的假设，通常也不需要任何形式关于密码的先验，这使得这些算法更为灵活。先前的工作集中于对用户使用的密码进行安全强度评估，模型化密码分布的过程基于传统方法，使用递归神经网络代替以往的马尔可夫模型，转移式地生成候选密码。Briland 等所提出的 PassGAN 直接对密码分布建模，以采样的方式生成候选密码，但是该算法无法进行定向的密码猜测攻击，在拥有攻击对象额外信息时无法进一步提升攻击效率。

2 正文

2.1 方法

2.1.1 对抗生成网络

对抗生成网络 (Generative Adversarial Networks) 基于严格的数学推导, 以新的方式构建了生成框架, 与深度学习模型结合后, 得到了拥有广泛适用性与出色生成性能的算法, 为深度学习开辟了新的研究领域。对抗生成网络的主要任务是从已知的数据样本 $\mathcal{S} = \{x_1, x_2, \dots, x_n\}$ 得到原始的数据分布, 并从其中采样生成新的样本。生成式算法在模型化的过程中, 一般依赖于解析形式的表达式, 因此大多数时候无法捕捉数据的复杂特征 [Murphy, 2012](#)。GANs 并非直接模型化数据分布, 而是对分布变换的过程进行建模, 它含有一个生成深度神经网络 G , 将来自正态分布或者均匀部分的多维随机样本 z 进行变换生成所需分布的一个采样。这个过程最大的问题在于如何评估生成深度神经网络 G 的表现, 即如何度量变换分布与真实分布的距离, GANs 将其转化为一个二分类问题, 更新 G 的参数依赖于一个判别深度神经网络 D , D 需要分辨接收到的采样是否来自原始数据分布, 而已知 \mathcal{S} 中的样本点是来自原始分布的, 这即给出了两者间距离的度量。如果使用更加规范的方式来描述这个过程, 可以被简单地表示为如下的极大极小博弈:

$$\min_{\theta_G} \max_{\theta_D} \sum_{i=1}^n \log f(x_i; \theta_D) + \sum_{j=1}^n \log (1 - f(g(z_j; \theta_G); \theta_D)) \quad (1)$$

这里 $f(x; \theta_D)$ 与 $g(z_j; \theta_G)$ 分别代表 D 与 G , 通过以上所述的极小极大博弈, 两个网络最终将逐渐达到均衡状态, 即 G 输出的采样相当接近原始数据分布, D 能够比较准确地区分采样是否来自原始数据分布。对抗生成网络由 Goodfellow 等 [Goodfellow et al., 2014](#) 提出, 目前对 GANs 有了许多改进的版本, 如 WGAN [Arjovsky, Chintala, and Bottou, 2017](#), IWGAN [Gulrajani et al., 2017](#) 等。在这项工作中, 针对现有基于深度学习的方法无法进行定向攻击的问题, 我们利用 CGAN [Mirza and Osindero, 2014](#) 对其进行了改进, 这使得我们的方法能够利用关于受攻击者的额外信息。

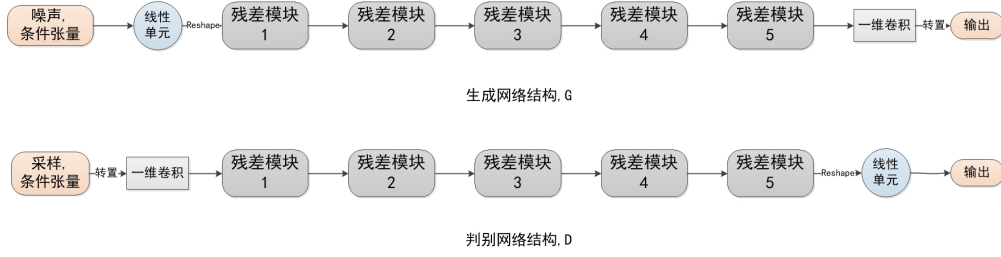


Figure 2.1: TPGGAN 的结构，生成网络与判别网络都可以输入额外的条件张量，对生成过程进行一定程度的限制。

2.1.2 TPGGAN

基于 Briland 等 [Hitaj et al., 2017](#) 的工作，我们对生成网络与判别网络的输入做了一些调整，使得我们的方法能够更加灵活地利用额外补充的信息，图 2.1 给出了方法的整体结构。

CGAN 在 Mehdi 等 [Mirza and Osindero, 2014](#) 的工作中，作者指出对抗生成网络可以被扩展为一个条件模型，通过同时对生成网络和判别网络加入某个额外信息 \mathbf{y} ，以完成条件化的过程。 \mathbf{y} 可以是任何形式的辅助信息，比如类别信息或者来源与其它模式的数据。此时目标函数将呈现如下的形式：

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{y})))] \quad (2)$$

IWGAN 在实例化对抗生成网络的过程中，我们使用了 Ishaan 等 [Gulrajani et al., 2017](#) 的方式。尽管 GANs 本身提供了强大的生成模型，但其训练过程相当不稳定，模型难以收敛。Martin 等 [Arjovsky, Chintala, and Bottou, 2017](#) 所提出的 Wasserstein GAN 使得 GANs 的训练更加稳定，但是在许多情况只能生成退化的样本或者无法收敛，Ishaan 等使用梯度范数衰减的方式替代 WGAN 的权重剪枝过程，使得 IWGAN 相比 WGAN 的训练过程更加简单稳定。

WGAN 通过 Kantorovich-Rubinstein 对偶的方式进行如下的优化：

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_r} [D(\mathbf{x})] - \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathbb{P}_g} [D(\tilde{\mathbf{x}})] \quad (3)$$

这里 \mathcal{D} 是 1-Lipschitz 函数， \mathbb{P}_g 是由 $\tilde{\mathbf{x}} = G(\mathbf{z})$ ， $\mathbf{z} \sim p(\mathbf{z})$ 隐式定义模型分布。在拥有一个最优判别器的情况下（由于在 WGAN 中并不是用于分类，在这里它被称为 critic），对上述的损失函数进行优化，使得生成器的参数能够极小化 $W(\mathbb{P}_r, \mathbb{P}_g)$ 。

在 WGAN 中为了增强对 critic 的 Lipschitz 限制，作者提出将网络的权重进行裁剪，

使得权重在闭区间 $[-c, c]$ 内。Ishaan 的工作表明，这样的方式将会导致在某些情况网络最终的退化，即变为一些简单函数。作者使用梯度惩罚的方式，来替代原有工作中增强 Lipschitz 限制的策略。由于可微函数为 1-Lipschitz 当且仅当其梯度范数处处至多为 1，所以这里直接在 critic 输出对输入梯度上进行范数惩罚，进行替换后目标函数如下：

$$L = \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathbb{P}_g} [D(\tilde{\mathbf{x}})] - \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_r} [D(\mathbf{x})] + \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim \mathbb{P}_X} [(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\|_2 - 1)^2] \quad (4)$$

与以往目标函数不同之处在于，添加了对 critic 梯度的正则项，使得优化朝着梯度范数趋于 1 的方向进行，实验上表明进行双向的惩罚通常最终效果更好（作者没有选择使梯度范数不超过 1，即单边惩罚）。

同时，为了实现条件化的生成过程，最终 TPGGAN 的优化函数可表示如下：

$$L = \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathbb{P}_g} [D(\tilde{\mathbf{x}}|\mathbf{y})] - \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_r} [D(\mathbf{x}|\mathbf{y})] + \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim \mathbb{P}_X} [(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\|_2 - 1)^2] \quad (5)$$

ResNet 为了解决深度神经网络在层数过深时难以优化、深层退化的问题，He 等 [He et al., 2016](#) 提出了深度残差网络，在层之间以残差的形式进行模型化。具体来说，它以如下的方式进行前向传递：

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x} \quad (6)$$

在参数化的过程中，网络的学习对象为层之间的残差，这保证网络的每一层都是有意义的，即便发生梯度消失的问题，也可以保证网络进行前向传递时至少为一个恒等映射。Weinan 等 [Weinan, 2017](#) 的工作表明，深度残差网络可以被理解为一个离散动力系统，并以此为基础对深度残差网络做了一些解释：

$$\begin{aligned} y_l &= h(z_l) + F(z_l, W_l) \\ z_{l+1} &= g(y_l) \end{aligned} \quad (7)$$

其中 Z_l 和 Z_{l+1} 为第 l 层的输入和输出， y_l 为第 l 层的辅助变量， h 和 g 为某个映射，可以为线性或者非线性的。在深度神经网络中，从实践经验中看，如果 g 和 h 近似于恒等映射，训练将会很顺畅。这可以被解释如下，如果令 G 为 g 的逆映射，则可以将上述动力系统写为：

$$z_{l+1} = G(h(z_l) + F(z_l, W_l)) \quad (8)$$

为了保持系统的梯度稳定性，上述方程右边的梯度需要接近于恒等映射。这也就是说梯度由后向前传时，在范数上需要保持稳定，如果令 h 和 g 都为恒等映射，那么有 $\nabla G \nabla h \sim I$,

F 作为较小的随机扰动，这样梯度的传递就非常平稳。

但是如果 h 和 g 就是恒等映射，那么式子 8 就可以写为：

$$z_{l+1} = z_l + F(z_l, W_l) \quad (9)$$

这正是 He 等工作中提出的深度残差网络的形式，并且上述的式子可以看作如下的连续动力系统的离散化：

$$\frac{dz}{dt} = F(z, W(t)) \quad (10)$$

这里 t 相当于层级 l 的连续化，可以说残差网络即式 9 正是连续动力系统 10 的欧拉离散化，如果从导数定义的角度来观察式子 10，当 t 的变化趋向于无穷小时，隐藏状态的变化 dz 可以通过神经网络进行表示，而 t 变化到终止层时， $z(t)$ 的变化即为一次完整的前向传播，深度残差网络本质上可以解释为动力系统，并能够使用微分方程进行表示。

2.2 实验

2.2.1 数据集

RockYou Dataset RockYou 数据集 [RockYou 2010](#) 包含 32503388 个密码样本，在实验过程中，我们使用少于十个字符的密码（这部分含有 29599680 个样本，占据数据集超过 90%），并使用其中的 80% 作为训练集合对各个密码猜测工具进行训练，并对剩下的 20% (5919936 个样本) 进行筛选，确保没有在训练集中出现过，最终得到 1978367 个密码样本作为测试集。

LinkedIn Dataset LinkedIn 数据集 [LinkedIn](#) 同样也被用于测试过程中，我们对其进行相同的处理，并确保其中的样本异于从 RockYou 选定的训练集。它总共包含 60065486 个唯一密码样本，其中有 43354871 字符长度不超过 10，最终的测试集包含 40593536 个样本（去除在训练集中出现过的样本）。

2.2.2 实验组成

我们首先对 TPGGAN 的密码猜测性能进行评估，在训练集上完成训练过程后，进行候选密码生成过程，并与测试集中的密码样本进行匹配，作为对比的方法包含：FLAMelicher et al., 2016，3-gram 形式的马尔可夫模型 Dorsey, 2017，PCFGs Weir, 2009，使用规则攻击的 JtRSPIDERLABS, 2012 与 HashCat HashCat 2017，以及 PassGAN Hitaj et al., 2017，并

结合 TPGGAN 与 HashCat Best64 两种方法进行测试。其次，我们测试了 TPGGAN 在定向攻击时的效率，即完成训练后抽取测试集子集，将子集中每个样本的部分字符作为额外信息输入 TPGGAN 中，与完全不提供任何额外信息的情况进行效率上的对比。

2.2.3 非定向攻击

Table 2.1: TPGGAN 产生的唯一密码数量，以及与 RockYou 测试集中样本（1978367 个）的匹配率

密码生成数量	唯一密码数量	匹配率
10^4	9,934	0.005%
10^5	96,720	0.049%
10^6	846,723	0.376%
10^7	7,035,586	2.141%
10^8	53,001,214	6.732%
10^9	352,684,864	16.012%
10^{10}	2,231,742,374	25.975%
10^{11}	9,324,676,543	37.324%

在表格 2.1 中，我们展示 TPGGAN 在不同测试数量下，所生成的唯一候选密码数量，以及对应情况下在 RockYou 测试集中的匹配率。可以看出随着密码生成数量的增加，TPGGAN 所覆盖的集合大小在不断增加，匹配率也以较大幅度上升。经过实验测试，在生成数量量级为 10^{10} 时，匹配率的增长率逐渐变缓，同时所生成唯一密码的数量增长幅度降低，在生成过程中产生了大量的重复密码样本，密码生成数量为 10^{11} 时，生成更多样本已经难以提升匹配率。

Table 2.2: TPGGAN 在 RockYou 测试集上达到各个方法相同匹配率所需生成样本数量

方法	唯一密码	匹配率	TPGGAN 所需生成样本数	TPGGAN 匹配率
JTR(Spyderlab)	10^9	23.32%	$1.2 \cdot 10^9$	23.32%
3-gram 马尔可夫模型	$4.9 \cdot 10^8$	26.93%	$2.51 \cdot 10^9$	26.93%
HashCat(gen2)	10^9	30.22%	$4.8 \cdot 10^9$	31.60%
HashCat(Best64)	$3.6 \cdot 10^8$	31.84%	$5.04 \cdot 10^9$	31.85%
PCFG	10^9	24.59%	$1.9 \cdot 10^9$	25.12%
FLA($\rho = 10^{-10}$)	10^9	23.32%	$1.2 \cdot 10^9$	23.32%
PassGAN	10^{10}	26.036%	$1.1 \cdot 10^{10}$	26.041%

在表格 2.2 与表格 2.3 中，我们着重于验证 TPGGAN 是否能达到其它密码猜测工具的性能，尽管在算法的构建过程中没有加入任何形式的先验知识，而与之对比的方法通常来自长时间更新迭代的人为规则设计并结合了马尔科夫模型。结果表明 TPGGAN 完全可以达到现有工具的匹配率，但另一方面，TPGGAN 通常需要多产生一个数量级的密码样本

Table 2.3: TPGGAN 在 LinkedIn 测试集上达到各个方法相同匹配率所需生成样本数量

方法	唯一密码	匹配率	TPGGAN 所需生成样本数	TPGGAN 匹配率
JTR(Spyderlab)	10^9	16.85%	$2.5 \cdot 10^9$	16.85%
3-gram 马尔可夫模型	$4.9 \cdot 10^8$	14.36%	$1.7 \cdot 10^9$	14.36%
HashCat(gen2)	10^9	15.54%	$2.2 \cdot 10^9$	15.54%
HashCat(Best64)	$3.6 \cdot 10^8$	17.67%	$3.5 \cdot 10^9$	18.24%
PCFG	10^9	17.95%	$3.6 \cdot 10^9$	18.24%
FLA($\rho = 10^{-10}$)	$7.4 \cdot 10^8$	20.42%	$5.8 \cdot 10^9$	20.57%
PassGAN	10^{10}	24.11%	$9 \cdot 10^9$	24.11%

才能达到相同的匹配率，在 RockYou 与 LinkedIn 测试集上都出现了同样的结果，这表明了在特征表示的过程中，TPGGAN 学习到的特征表示区分度不足，与回归神经网络以及马尔可夫模型这种概率图形式的模型相比，生成模型通常需要对整个分布进行建模，这导致了内在的混淆性。

与 PassGAN 的对比表明，对 PassGAN 的条件化不会导致模型性能的衰减，在一些情况甚至会一定程度地提升原有模型的性能。并且之后的实验将会表明，条件化的模型在拥有额外信息时，其猜测效率将会得到较大幅度的提升，这是原有模型所不具备的特点。

TPGGAN 与 HashCat 结合基于规则的方法，TPGGAN 将能够以更快的速度生成新的密码样本，而这在实验中通常需要大量的采样才能达到同样的效果。为了评估 TPGGAN 结合 HashCat 后的性能，我们首先移除了 RockYou 以及 LinkedIn 测试集中 HashCat Best64（这是实验中在 RockYou 测试集上表现最好的方法）已经匹配到的密码，这样我们得到了一个新的测试集合，包含 1348300 来自 RockYou 的样本以及 33394178 个 LinkedIn 的样本。

我们的实验结果表明，随着 TPGGAN 生成的样本增加，我们能够不断提高 HashCat 在新测试集上的匹配率，虽然仅使用 HashCat 本身无法继续提升这个数值。具体来说，使用 $7 \cdot 10^9$ 由 TPGGAN 产生的密码，HashCat 能够匹配 54% 新的 RockYou 测试集中的密码，以及 72% 的新 LinkedIn 中的密码。这表明 TPGGAN 对原始密码分布进行建模时，所捕捉到的特征与人为设计规则是有相当差异性的，TPGGAN 使我们能够探索以往无法发现的新的规则，与以往方法结合后，能够显著地提升算法性能。

2.2.4 定向攻击

定向攻击通常指攻击者对某个特定受攻击者的为某项服务设置的密码而进行的密码猜测攻击，这大部分是由对受攻击的个人信息，以及他其它帐号所泄漏出的密码或者一些个人可认证信息进行分析进行的。这在实际情况中通常是十分有效的，并且也是一个

较为严重的安全问题，攻击者可以较为容易地获得许多形式的个人可验证信息，也可以由因为违规行为而不断泄漏的数据中获得密码样本。举例来说，近期有一次大规模的个人可验证信息泄漏发生在 2016 年四月 [Turkey: personal data of 50 million citizens leaked online 2016](#)，包含有 5 千万土耳其国民的个人信息，占总人口的 64%。根据 CNNIC2015 的报道 [Nearly 80 percent of Internet users suffer identity leaks 2015](#)，超过 78.2% 的近七亿中国网络用户曾遭受过个人可验证信息泄漏，由于几次违规数据泄漏，超过两亿五千万美国网络用户成为个人可验证信息与密码泄漏的受害者 [Four Years Later, Anthem Breached Again: Hackers Stole Credentials 2015](#)。尽管获取受攻击用户的信息现在已经不成为一件困难的事情，但是定向攻击仍然具备相当的难度。

首先，用户使用的密码之间常常拥有较大的差异性。当创建密码时，一些用户会重复使用一个已存在的密码，而另一些人会对其进行一定的修改。一些用户会将个人可验证信息加入他们的密码中，但其它的人则完全没有这种习惯。有些偏向于使用数字，有些更常使用符号，诸如此类。因此，用户群体针对一个网络服务所创建的密码相互之间会有着很大差异。

其次，用户的个人可验证信息是相当多样的。如姓名爱好等由字符组成，生日、电话号码由数字组成。一些个人可验证信息可以被直接融入密码中，比如姓名生日的信息，而性别和教育这些信息难以被直接使用。而在 Ding 等 [Wang et al., 2016](#) 的工作中，这些不能直接用于密码设计中的信息，通常对用户选择密码时有着一定的影响。

最后，当用户重复利用已有密码时，他们的变换规则拥有很大的差异性。当给定一个密码时，一般存在超过十个变换格则，比如插入、删除、大写以及 leet 等，都会被用户利用于创建新的密码。如何以模型化每个个体用户所使用规则并不是一件容易的事。

TPGGAN 并不掌握任何形式的先验知识，在模型化用户利用自身信息或者已有密码设计新密码时，我们的方法是完全基于条件化模型的思想来进行这个过程的。在这部分实验中，我们以较为直观的形式展示了 TPGGAN 进行定向攻击时的算法性能。将测试集中的每个测试样本的一部分字符取出，生成一个条件张量，并输入 TPGGAN 中进行候选密码生成。如果能够以更少的生成数达到相同的匹配数，那么即表明条件张量确实提升了 TPGGAN 的攻击效率，即区别于非定向攻击，从简易起见，我们将使用了条件张量的 TPGGAN 记为 TPGGAN-C，在这里我们固定提供测试样本的四分之一字符。

在表格 2.4 中，TPGGAN-C 在拥有一部分测试样本的信息后，其匹配率有了 35% 以上的提升，最高匹配率可以达到 50.581%，这是非定向攻击中无法达到的匹配率。这也表明了，通过将生成模型进行条件化，可以使得模型的生成过程得到一定的限制，具体表现在 TPGGAN 则为我们可以通过提供受攻击者的一定信息，以较大程度地提升算法进行密码猜测的效率，在这里我们直接提供了密码的一部分字符。在之后的工作中，我们可以尝试

Table 2.4: 在固定生成数的情况,TPGGAN 和 TPGGAN-C 在 RockYou 上的匹配率

生成数	TPGGAN 匹配率	TPGGAN-C 匹配率	提升率
10^7	2.141%	2.912%	36%
10^8	6.732%	10.839%	61%
10^9	16.012%	24.658%	54%
10^{10}	25.975%	36.885%	42%
10^{11}	37.324%	50.581%	35%

更为贴合实际的定向攻击形式，即提供用户的个人可验证信息或者曾经使用密码来作为额外的条件张量，可以预见这部分额外信息同样会带来算法效率上的提升。

2.3 评论

在这一节中，我们主要总结实验部分中得到的一些结果，并不限于展示实验结果。

在本质上，对抗生成模型对高斯分布到真实数据分布的变换进行了建模，这在过去是难以进行的。相关的数学理论是相当丰富的，但并不存在实践性能优秀、便于实现、对大规模数据有较高效率的算法。Goodfellow 的工作从理论出发，证明了一个性能优秀的函数近似器可以通过进行一个极大极小博弈完成上述的任务，从而为深度学习领域揭开了新的篇章，也为我们展示了许多可能性。

正如我们所强调的，TPGGAN 并不需要任何形式有关密码的先验知识，相比现有的常用密码猜测工具，它是相当廉价易得的，不需要长时间人为的更新迭代就可以通过大量生成密码样本的方式，达到相同的匹配率。现有的基于规则的密码猜测算法是高效的，同时也是十分受限的。在我们的实验中，这一点由 TPGGAN 与 HashCat 进行结合表现，尽管 HashCat 经过了长时间的专家改进，它仍然倾向于真实密码分布中的较小部分采样，一些密码的特征它仍然没有捕捉到，当对 TPGGAN 的生成结果使用规则变换算法时，其匹配被较大幅度提升了。

在定向密码猜测攻击这个任务上，现有的方法相当有限，以往的非定向攻击猜测工具难以扩展到新的任务上，即便获取了一些受攻击用户的信息，以往的方法也无法对其进行有效的利用。而生成模型则拥有更好的扩展性，通过对模型的目标函数进行条件化，我们就能够利用额外的信息，对生成的密码样本进行一定的限制。从实验结果上来看，这样的做法是相当有成效的，新增的条件张量很大程度地提高了方法的效率，这是以往方法所不具备的特点。

在实验中，我们也可以看到 TPGGAN 的不足之处。超过以往方法的匹配率时，通常需要多一个数量级的密码样本生成。显然，这很大程度上是由于在生成过程中，产生了大量的重复候选密码。通过增加模型的容量，即加深网络的深度或者说增加模型的参数量，

我们可以较大程度上保证所重构的模型是相当接近真实密码分布的，然而这样的方式也不能直接解决上述效率不足的问题。从另一方面来说，大量的高斯分布采样被映射到相同的输出结果，我们可以在输入采样时，对采样间的区分度做一定限制，更为重要的应当在模型进行特征表示时，增强不同表示间的区分程度，以确保生成样本的多样性。

3 结论

在这篇文章中，我们提出了 TPGGAN，这是第一个基于对抗生成网络并拥有定向攻击能力的密码猜测算法。我们希望 TPGGAN 能够从密码数据集中端到端的学习到真实密码的分布，而不需要任何形式的先验知识。从结果上看，TPGGAN 以自治的形式形成了一个拥有同等猜测性能的算法，相较马尔科夫模型以及概率上下文无关语法的方法，TPGGAN 对分布本身没有较强的假设，算法生成密码过程也不限于简单的规则变换，而是以从模型化的分布中采样得到的。

我们在两个较大的密码数据集上对 TPGGAN 以及现有的方法进行了验证，以测试其生成密码猜测的性能。我们的结果表明了相比现有最好的方法，TPGGAN 展现了很高的竞争力，尽管 TPGGAN 需要进行更多的尝试数，它始终能够与现有的方法拥有相当的密码匹配率。

TPGGAN 与以往方法最大的不同之处在于，我们的方法能够有效地利用额外的用户信息，可以被用于针对特定用户的定向攻击中。非定向攻击通常用于离线密码修复中，在这种场景中，攻击者希望尽可能多地匹配哈希与文本密码，而并不关注某个特定用户群体。而在真实的攻击场景下，受攻击用户一般拥有不同的攻击价值，攻击者更希望首先破解那些更具价值的用户密码，一旦攻击者确定了攻击对象，其攻击方式就将会改变。这是因为我们在正文中所说的，用户个人信息以及曾使用密码泄漏的情况是十分常见的，攻击者拥有很多途径可以较简单地获取这些信息，这时定向攻击算法就成为必要的。在实验中，我们的方法展现出了较好的利用额外条件信息的能力，可以将猜测效率提升 35% 以上，这是十分显著的。

在实验部分我们讨论了 TPGGAN 的优势所在，也指出了其效率不足的问题，并讨论了它提升效率的可能性，这部分工作还需要更多的尝试，我们会将其放在未来的工作中。同时，TPGGAN 拥有着更多的潜力，它可以作为生成 honeywords [Juels and Rivest, 2013](#) 的工具，honeywords 是一些诱饵密码，如果将其混入真实的密码数据集中，那么它会很大程度地降低密码数据集作为学习样本的价值，这可以作为一种非主动的保护数据库的机制，这也被考虑在我们的未来工作中。

4 参考文献

- [1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein gan”. In: *arXiv preprint arXiv:1701.07875* (2017).
- [2] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. “Adaptive Password-Strength Meters from Markov Models.” In: *NDSS*. 2012.
- [3] Angelo Ciaramella et al. “Neural network techniques for proactive password checking”. In: *IEEE Transactions on Dependable and Secure Computing* 3.4 (2006), pp. 327–339.
- [4] Matteo Dell’Amico, Pietro Michiardi, and Yves Roudier. “Password strength: An empirical analysis”. In: *2010 Proceedings IEEE INFOCOM*. IEEE. 2010, pp. 1–9.
- [5] Brannon Dorsey. *Markov-chain password generator*. <https://github.com/brannondorsey/markov-passwords>. 2017.
- [6] *Four Years Later, Anthem Breached Again: Hackers Stole Credentials*. <http://t.cn/RqWrMKC>. 2015.
- [7] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.
- [8] Ishaan Gulrajani et al. “Improved training of wasserstein gans”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 5767–5777.
- [9] *HashCat*. <https://hashcat.net>. 2017.
- [10] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.
- [11] Briland Hitaj et al. “Passgan: A deep learning approach for password guessing”. In: *arXiv preprint arXiv:1709.00440* (2017).
- [12] *John the Ripper*. <https://www.openwall.com/john/>. 2017.
- [13] Ari Juels and Ronald L Rivest. “Honeywords: Making password-cracking detectable”. In: *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM. 2013, pp. 145–160.

- [14] Patrick Gage Kelley et al. “Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms”. In: *2012 IEEE Symposium on Security and Privacy*. IEEE. 2012, pp. 523–537.
- [15] *LinkedIn*. <https://hashes.org/public.php>.
- [16] Jerry Ma et al. “A study of probabilistic password models”. In: *2014 IEEE Symposium on Security and Privacy*. IEEE. 2014, pp. 689–704.
- [17] William Melicher et al. “Fast, lean, and accurate: Modeling password guessability using neural networks”. In: *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 2016, pp. 175–191.
- [18] Mehdi Mirza and Simon Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).
- [19] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.
- [20] Arvind Narayanan and Vitaly Shmatikov. “Fast dictionary attacks on passwords using time-space tradeoff”. In: *Proceedings of the 12th ACM conference on Computer and communications security*. ACM. 2005, pp. 364–372.
- [21] *Nearly 80 percent of Internet users suffer identity leaks*. <http://bit.ly/2b9TEdn>. 2015.
- [22] *RockYou*. <http://downloads.skullsecurity.org/passwords/rockyou.txt.bz2>. 2010.
- [23] TRUSTWAVE SPIDERLABS. *KoreLogic-Rules*. <https://github.com/SpiderLabs/KoreLogic-Rules>. 2012.
- [24] *Turkey: personal data of 50 million citizens leaked online*. <http://bit.ly/1TPA4j4>. 2016.
- [25] Ding Wang et al. “Targeted online password guessing: An underestimated threat”. In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. ACM. 2016, pp. 1242–1254.
- [26] E Weinan. “A proposal on machine learning via dynamical systems”. In: *Communications in Mathematics and Statistics* 5.1 (2017), pp. 1–11.
- [27] Matt Weir. *Probabilistic Password Cracker*. https://sites.google.com/site/reusablesec/Home/password-cracking-tools/probablistic_cracker. 2009.

- [28] Matt Weir et al. “Password cracking using probabilistic context-free grammars”. In: *2009 30th IEEE Symposium on Security and Privacy*. IEEE. 2009, pp. 391–405.
- [29] Roman V Yampolskiy. “Analyzing user password selection behavior for reduction of password space”. In: *Proceedings 40th Annual 2006 International Carnahan Conference on Security Technology*. IEEE. 2006, pp. 109–115.